computation

MDPI

Article

Enhancing Computational Precision for Lattice Boltzmann Schemes in Porous Media Flows

Farrel Gray [1] and Edo Boek [1,2,*]

[1] Qatar Carbonates and Carbon Storage Research Centre (QCCSRC), Department of Chemical Engineering, South Kensington Campus, Imperial College London, London SW7 2AZ, UK; farrel.gray09@imperial.ac.uk
[2] Department of Chemistry, University of Cambridge, Lensfield Road, Cambridge CB2 1EW, UK
* Correspondence: esb30@cam.ac.uk

Academic Editors: Qinjun Kang and Li Chen
Received: 1 December 2015; Accepted: 9 February 2016; Published: 17 February 2016

Abstract: We reassess a method for increasing the computational accuracy of lattice Boltzmann schemes by a simple transformation of the distribution function originally proposed by Skordos which was found to give a marginal increase in accuracy in the original paper. We restate the method and give further important implementation considerations which were missed in the original work and show that this method can in fact enhance the precision of velocity field calculations by orders of magnitude and does not lose accuracy when velocities are small, unlike the usual LB approach. The analysis is framed within the multiple-relaxation-time method for porous media flows, however the approach extends directly to other lattice Boltzmann schemes. First, we compute the flow between parallel plates and compare the error from the analytical profile for the traditional approach and the transformed scheme using single (4-byte) and double (8-byte) precision. Then we compute the flow inside a complex-structured porous medium and show that the traditional approach using single precision leads to large, systematic errors compared to double precision, whereas the transformed approach avoids this issue whilst maintaining all the computational efficiency benefits of using single precision.

Keywords: lattice Boltzmann; porous media; precision

1. Introduction

The lattice Boltzmann (LB) method solves a discrete, meso-scale form of the Boltzmann equation [1], which can be shown to reduce to the incompressible Navier-Stokes equation in the low Mach number limit. Because of its computational efficiency and simplicity, the LB approach is now widely-used in many fields of computational fluid dynamics such as thermal and multiphase flows [2] and reactive transport [3], while rapidly developing in applications such as turbulent flows [4].

In a 1993 paper, Skordos presented an approach to increase the computational precision of lattice Boltzmann schemes in which the distribution functions were transformed by negating the equilibrium zero-velocity distribution function [5]. In principle, this should have maintained more significant bits during the calculation, leading to higher accuracy which no longer depended on the local velocity. Curiously though, when compared with the standard method of calculation involving the unmodified distribution functions, the approach was shown only to provide a very minor benefit to the accuracy of the calculation, which was still strongly dependent on the velocity (see Figure 4 in [5]).

In this work, we revisit the idea and show that application of the method with an important extra consideration can indeed lead to orders of magnitude increase in the accuracy of LB calculations while incurring no extra computational cost. The velocity-dependence of the accuracy, which is observed with the standard approach, is also removed. In the first section, we describe the method and highlight important considerations in the implementation that were missed in Skordos' original paper

which probably account for the underwhelming result. Then we demonstrate in a simple capillary how the accuracy of the solution is greatly improved, and unfavourable velocity-dependence of the accuracy is removed. Finally, the method is applied to the calculation of single-phase flow in a complex porous medium and shown to be critical to obtaining the correct flow-field if single precision is to be used. Many researchers have incorporated this method into their codes in recent years, including the open-source codes Palabos [6] and OpenLB [7]. Dellar, for example, also used Skordos' approach [8] but, to the best of our knowledge, the full potential of this method has not been shown or quantified in the literature. This paper is intended to serve as a concise manual which will help LB practitioners fully exploit this simple but highly effective method.

The performance of optimised implementations of the LB model are known to be bound by memory bandwidth on CPUs and graphics processing units (GPUs). Using the single precision (float) datatype rather than double precision can provide up to twice the throughput between the main memory and the cache, offering a comparable performance enhancement. Cache occupancy is also doubled, resulting in fewer cache-misses. Single-precision arithmetic itself is faster than double precision in GPUs, and can be on CPUs if compiled using optimal hardware instructions. The performance benefits of using single precision rather than double precision are greatly advantageous in multi-component or reactive flow simulations which may take days to run, as well as the reduced memory requirement, which is often a limiting factor on modern GPUs.

Here, we apply LB to single-phase flow calculations in a porous medium, such as is used for permeability and transport calculations in petro-physical analysis, as a practical example. Pore structures in certain carbonate rocks can be extremely heterogeneous [9] and as such lead to wide flow-velocity distributions. When simulating flow in these cases, it is important not to have velocity-dependence in the accuracy of the calculation as low velocity regions are found to play an important role in transport properties [10,11].

We use an LB model which is particularly suited to porous media flows in this work. Although the Bhatnagar-Gross-Krook (BGK) approximation of the collision operator in conjunction with the halfway bounce-back scheme for treating solid-fluid boundaries is most commonly found in the literature, this model suffers from deficiencies such as viscosity-dependent slip at the walls [12] and low numerical stability. As such, it is difficult to obtain the true flow properties of a porous medium with this method. Instead, the multiple-relaxation-time (MRT) model offers much greater flexibility by transforming the distribution function into a set of moments, each of which may be relaxed with an individual rate [13,14]. By tuning the unphysical relaxation parameters, viscosity-independence can be achieved with the bounce-back boundary conditions [15]. Although boundary interpolation schemes demonstrate slightly more consistent behaviour [16,17], we have found that the standard bounce-back method gives accurate results for flow in complex porous media [18] and remain with this approach here for its computational efficiency. The fluid is driven by a body-force term, for which a precise treatment is needed to eliminate error terms in velocity gradients [19] and is incorporated into the MRT framework [20,21].

2. Method

The multiple-relaxation-time (MRT) calculation proceeds through the following steps.

Firstly, the macroscopic node variables density ρ and velocity $\boldsymbol{u}$ are obtained from the distribution function $\boldsymbol{f}(\boldsymbol{x}, t)$

$$\rho = \sum_i f_i \qquad \rho \boldsymbol{u} = \sum_i \boldsymbol{e}_i f_i + \frac{\rho \boldsymbol{g}}{2} \tag{1}$$

The definition for velocity is according to the forcing scheme of Guo [21] where $\boldsymbol{g}$ is a body-force, and $\boldsymbol{e}_i$ is the ith of the 19 velocity vectors at each node (Figure 1) which are defined as

$$\boldsymbol{e} = c \begin{bmatrix} 0 & 1 & -1 & 0 & 0 & 0 & 0 & 1 & -1 & 1 & -1 & 0 & 0 & 0 & 0 & 1 & -1 & 1 & -1 \\ 0 & 0 & 0 & 1 & -1 & 0 & 0 & 1 & 1 & -1 & -1 & 1 & -1 & 1 & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & -1 & 0 & 0 & 0 & 0 & 1 & 1 & -1 & -1 & 1 & 1 & -1 & -1 \end{bmatrix}^T \tag{2}$$

where $c = dx/dt$ is the lattice speed. The lattice spacing dx and time-step dt are both unity.

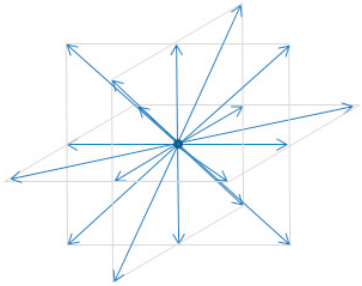

Figure 1. The 19 velocity components of the D3Q19 scheme.

The equilibrium distribution $\boldsymbol{f}^{Eq}(\rho, \boldsymbol{u})$ and forcing term $\boldsymbol{F}$ are then computed from the density and velocity of the node where the components are [21]

$$f_i^{Eq}(\rho, \boldsymbol{v}) = \rho w_i \left[1 + 3\boldsymbol{e}_i \cdot \boldsymbol{u} + \frac{9}{2}(\boldsymbol{e}_i \cdot \boldsymbol{u})^2 - \frac{3}{2}\boldsymbol{u}^2\right] \tag{3}$$

$$F_i = 3w_i \left[\boldsymbol{e}_i \cdot \boldsymbol{g} + \boldsymbol{u}\boldsymbol{g} : (3\boldsymbol{e}_i\boldsymbol{e}_i - \boldsymbol{I})\right] \tag{4}$$

Here w_i is the weight coefficient for the corresponding velocity vector, given by $w_0 = 1/3$, $w_{1-7} = 1/18$, $w_{8-18} = 1/36$ and $\boldsymbol{I}$ is the identity matrix.

The post-collision distribution function $\boldsymbol{f}'(\boldsymbol{x}, t)$ is computed using the MRT operator

$$\boldsymbol{f}'(\boldsymbol{x}, t) = \boldsymbol{f}(\boldsymbol{x}, t) + \boldsymbol{M}^{-1}\left[\boldsymbol{S}\boldsymbol{M}\left(\boldsymbol{f}^{Eq} - \boldsymbol{f}\right) + \left(\boldsymbol{I} - \frac{1}{2}\boldsymbol{S}\right)\boldsymbol{M}\boldsymbol{F}\right] \tag{5}$$

where $\boldsymbol{M}$ is the orthogonal matrix which transforms the distribution function into moment space and $\boldsymbol{M}^{-1}$ is its inverse. $\boldsymbol{S}$ is the diagonal matrix of relaxation rates for each moment. Full details of the MRT model can be found in [13]. Finally, the post-collisional distribution is streamed to the neighbouring nodes

$$f_i(\boldsymbol{x} + \boldsymbol{e}_i dt, t + dt) = f_i'(\boldsymbol{x}, t) \tag{6}$$

To illustrate how accuracy is lost when computing the algorithm numerically, we consider an expansion of the summations involved in computing the macroscopic velocity at the node

$$\rho\boldsymbol{u} = \begin{pmatrix} f_1 - f_2 + f_7 - f_8 + f_9 - f_{10} + f_{15} - f_{16} + f_{17} - f_{18} \\ f_3 - f_4 + f_7 + f_8 - f_9 - f_{10} + f_{11} - f_{12} + f_{13} - f_{14} \\ f_5 - f_6 + f_{11} + f_{12} - f_{13} - f_{14} + f_{15} + f_{16} - f_{17} - f_{18} \end{pmatrix} + \frac{\rho\boldsymbol{g}}{2} \tag{7}$$

Each component of the velocity is a summation of differences (subtractions). Physically this is the difference between vectors of opposing direction (Figure 1). If the node velocity is very small (and hence the distribution is quite symmetric), these differences can become smaller than the ability of the floating point data type to resolve, and the calculation becomes unstable.

Floating point types are stored as two parts: a mantissa (1.2666665) and exponent (-1) in the example

$$1.2666665 \times 10^{-1} \tag{8}$$

The mantissa in single precision (4-byte float) types is accurate to around 7 decimal places (in base 10), and the exponent ranges from −38 to +38. Therefore a difference can only be resolved to an accuracy of 10^{-7} the largest value in the negation. For a node with a small velocity, and distribution close to the zero-velocity (equilibrium) distribution $f^0(\rho_0 = 1, \boldsymbol{u}_0 = \mathbf{0})$, defined in D3Q19 by

$$f^0 = \rho_0 w_i = \begin{cases} 3.3333333 \times 10^{-1} & i = 0 \\ 5.5555556 \times 10^{-2} & i = 1 \; to \; 6 \\ 2.7777778 \times 10^{-2} & i = 7 \; to \; 18 \end{cases} \tag{9}$$

only velocities larger than $O(10^{-2}) \cdot 10^{-7} = 10^{-9}$ can be resolved at all, and the calculation is verifiably unstable when velocities are less than 10^{-7} since only a few decimal places accuracy is maintained.

When computing the flow in complex geometries, often with minimal connectivity, the magnitude of the velocity vectors can become smaller than this. A double precision implementation can handle this easily, but comes with the drawback of requiring considerably more compute time and memory than float precision. The small velocity can, in some cases, be counteracted by choosing a larger body-force. However if this is too large, it can exceed the model stability limits and the calculation will fail. Non-Darcy effects may also begin to appear in faster flow paths if the Reynolds number becomes too high [22]. In these cases, slow flows are desirable for accurate permeability calculation. Ideally, it should not become an art to choose an appropriate body-force for a given simulation domain; often the connectivity in a simulation is not known beforehand.

The calculation variables can be transformed so that greater accuracy is obtained. Instead of storing the distribution function $\boldsymbol{f}(\boldsymbol{x}, t)$, we define a perturbation $d\boldsymbol{f}$ from the zero-velocity distribution in the following way, as was suggested by Skordos [5]

$$\boldsymbol{f}(\boldsymbol{x}, t) = \boldsymbol{f}^0 + d\boldsymbol{f}(\boldsymbol{x}, t) \tag{10}$$

so that the distribution function at the node is decomposed into a reference distribution, chosen as the zero-velocity equilibrium distribution $\boldsymbol{f}^0(\rho_0, \boldsymbol{u}_0)$ with $\rho_0 = 1$ and $\boldsymbol{u}_0 = \mathbf{0}$, and a perturbation $d\boldsymbol{f}$. The appropriateness of this transformation rests on the lattice Boltzmann algorithm being memory-bandwidth-limited. This means that it would be preferable to compute the full distribution function locally by adding together $\boldsymbol{f}^0_{dbl}$ and $d\boldsymbol{f}_{flt}$ using double precision arithmetic (and where the subscripts *dbl* and *flt* express the variable's data type), storing $d\boldsymbol{f}$ as a single-precision (float) data type in the main memory. However, even this is unnecessary if we transform the macroscopic quantities

$$\rho = \sum_i f^0 + \sum_i df_i = 1 + \sum_i df_i \tag{11}$$

$$\rho \boldsymbol{u} - \frac{\rho \boldsymbol{g}}{2} = \sum_i \boldsymbol{e}_i df_i \tag{12}$$

Although we are still computing differences in the expression for velocity, the zero-velocity expression for the distribution $d\boldsymbol{f}$ is $\mathbf{0}$, so these differences are not bound by the order of magnitude of the calculation values, and are computed to full 7 d.p. accuracy. Note that the expression for the local density is still bound by the order 10^0.

The difference $\boldsymbol{f}^{Eq} - \boldsymbol{f}$ which arises in the collision term should be considered as well to maximise the accuracy of the calculation. Writing this in terms of the deviation distribution $d\boldsymbol{f}$, and using the subscripts *dbl* and *flt* to indicate the precision (double and single respectively) to which the variables might be stored or calculated, we obtain

$$\left(\boldsymbol{f}^{Eq} - \boldsymbol{f}\right)_{flt} = \boldsymbol{f}^{Eq}_{dbl} - \left(\boldsymbol{f}^0_{dbl} + d\boldsymbol{f}_{flt}\right)_{dbl} \tag{13}$$

The difference could be computed to double precision and converted back to single precision afterwards. This is because the node distribution and equilibrium distribution will be of order 10^{-1}, but the difference often considerably smaller.

Finally though, the need for double precision can be avoided completely if we first compute the equilibrium distribution as a perturbation from the zero equilibrium $d\boldsymbol{f}^{Eq} = \boldsymbol{f}^{Eq}(\rho, \boldsymbol{u}) - \boldsymbol{f}^{0}(\rho_0, \boldsymbol{u}_0)$ such that

$$df_i^{Eq} = \rho w_i \left[3\boldsymbol{e}_i \cdot \boldsymbol{u} + \frac{9}{2} (\boldsymbol{e}_i \cdot \boldsymbol{u})^2 - \frac{3}{2}\boldsymbol{u}^2 \right] + (\rho - \rho_0) w_i \tag{14}$$

The density difference $\rho - \rho_0$ will be small relative to the density values of order 10^0. However, this calculation can be made more accurate if we identify the following

$$\rho - \rho_0 = \sum_i df_i \tag{15}$$

We must explicitly compute this density difference from the right hand side summation and not via the densities themselves, otherwise precision is lost. This substitution is particularly important in the low velocity limit and was not mentioned in Skordos' original paper which only considered how the now comparable magnitudes of each term in (our) Equation (14) increases accuracy under addition, rather than the precision of each individual term [5]. Using the transformed quantities, the evolution of the LB equation is given as

$$d\boldsymbol{f}'(\boldsymbol{x}, t) = d\boldsymbol{f}(\boldsymbol{x}, t) + \boldsymbol{M}^{-1} \left[\boldsymbol{SM} \left(d\boldsymbol{f}^{Eq} - d\boldsymbol{f} \right) + \left(\boldsymbol{I} - \frac{1}{2}\boldsymbol{S} \right) \boldsymbol{MF} \right] \tag{16}$$

$$df_i(\boldsymbol{x} + \boldsymbol{e}_i dt, t + dt) = df_i'(\boldsymbol{x}, t) \tag{17}$$

In the single-relaxation-time (BGK) scheme, the collision part would similarly be written with a relaxation time τ

$$d\boldsymbol{f}'(\boldsymbol{x}, t) = d\boldsymbol{f}(\boldsymbol{x}, t) + \frac{1}{\tau} \left(d\boldsymbol{f}^{Eq} - d\boldsymbol{f} \right) + \left(1 - \frac{1}{2\tau} \right) \boldsymbol{F} \tag{18}$$

3. Results

The two methods of calculating the collision term are compared by computing the velocity distribution $U(x)$ between two infinite parallel plates of separation L (Figure 2), for which the analytical solution is well known, using a viscosity μ and body-force component g in a direction parallel to the plates:

$$U(x) = \frac{g}{2\mu} x (L - x) \tag{19}$$

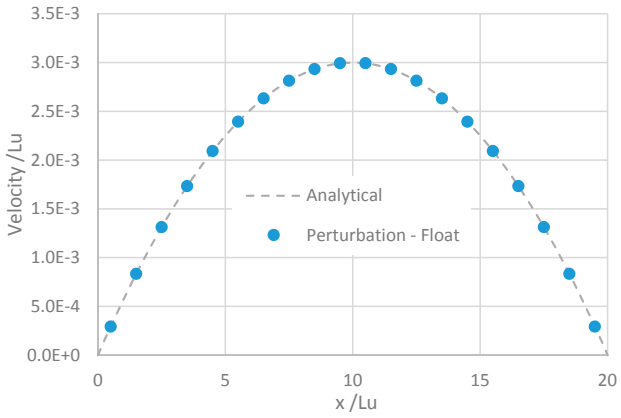

Figure 2. Poiseuille flow between two plates separated by 20 lattice units (Lu).

The method in which the complete distribution function is used, Equation (5) is referred to as the distribution method, and the scheme evolving the perturbation from the zero distribution, Equation (16) is referred to as the perturbation method. The deviation from the analytical solution for each method is computed as the relative error

$$\varepsilon(x) = \frac{|u(x) - U(x)|}{U(x)} \tag{20}$$

where $u(x)$ is the computed velocity component in the body-force direction as a function of position x between the plates.

For assuredly good convergence, computations are run for 10^6 time-steps. The error is shown in Figure 3 for the distribution and perturbation methods using respectively float (single-precision) data types and double (precision) data types. The body force used was $g = 10^{-6}$ in dimensionless lattice units (Lu) and the viscosity $\mu = \frac{1}{6}$ For this system, the perturbation methods are consistently 2 to 3 orders of magnitude more accurate than the distribution method of the same data type, though the perturbation method with float precision cannot match the accuracy of the double-precision distribution method in this calculation.

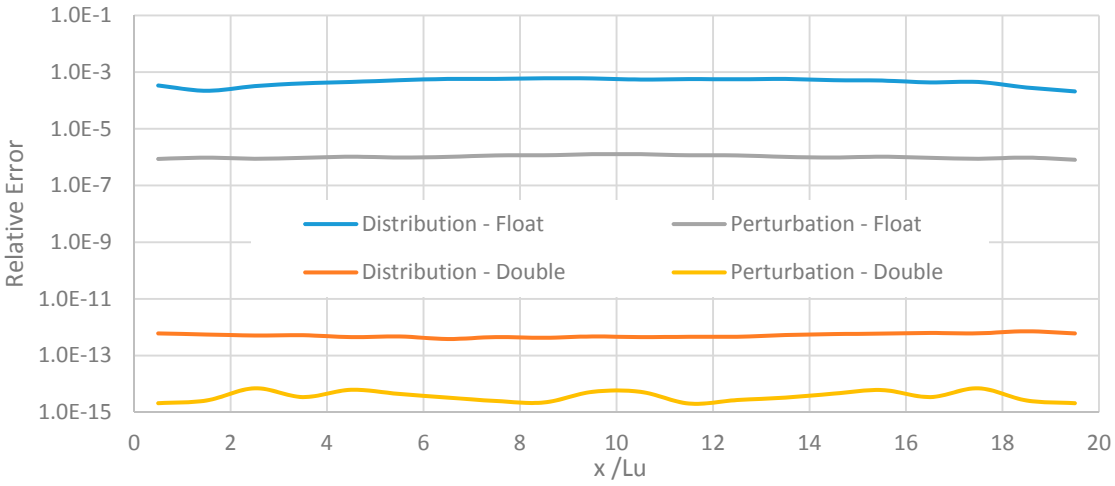

Figure 3. The relative error of the flow between plates for different calculation schemes and precisions for a body-force of 10^{-6}.

To illustrate how the accuracy of the perturbation scheme is freed from dependence on the magnitude of the velocities, the relative error averaged across the flow profile is plotted in Figure 4 for different average flow velocities. These were obtained by systematically reducing the body-force. It is clear that as the distribution methods lose accuracy for lower lattice velocities, the perturbation methods maintain a consistent relative error. We also note that the accuracy of the distribution methods converge to those of the perturbation schemes as the magnitude of the velocities approaches that of the zero equilibrium distribution Equation (9). Finally, the accuracy of the perturbation method with float precision can exceed that of the distribution method with double precision at grid velocities below around 10^{-11}.

To demonstrate the practical advantages of the enhanced accuracy afforded by the perturbation scheme at low velocities, we compute the flow in a 3D pore space image of a Portland carbonate rock sample (Figures 5 and 6) obtained from micro-CT imaging [10]. The sample is 400^3 lattice units in size, but reflected about the $x = 0$ plane to give continuous loop boundary conditions so that a body-force may be used [10]. The structure of the pore-space is highly heterogeneous and of low permeability. The body-force used was again $g_x = 10^{-6}$ and the viscosity $\mu = \frac{1}{6}$.

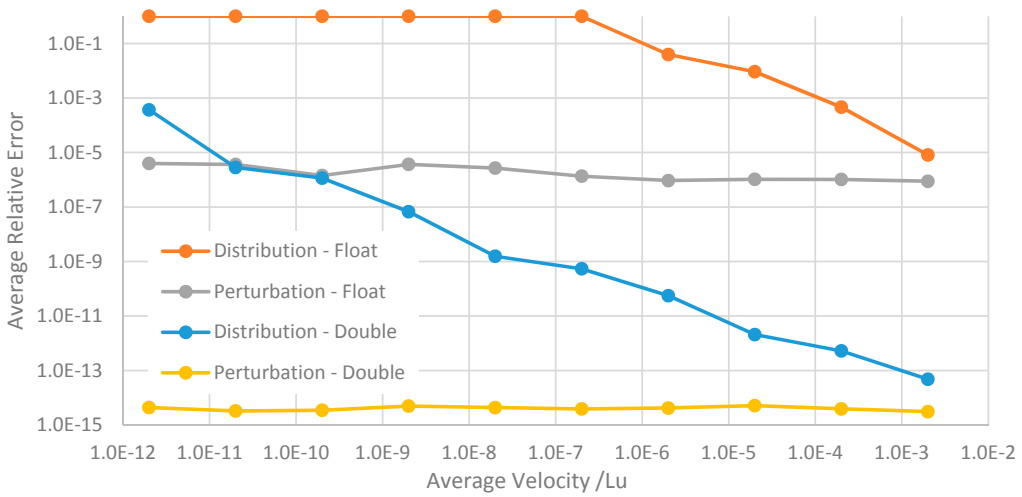

Figure 4. The average relative error of the flow profile between plates for different calculation schemes against mean flow velocity.

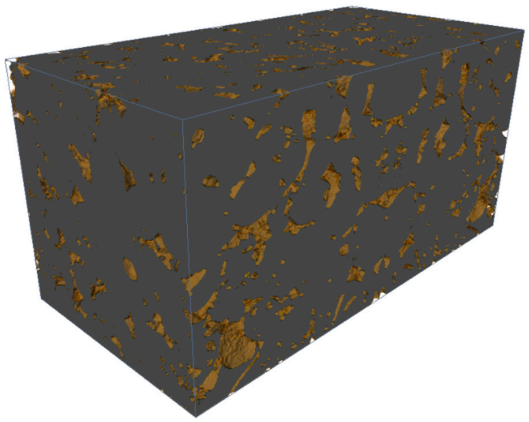

Figure 5. A sample of Portland carbonate of porosity 9.0%.

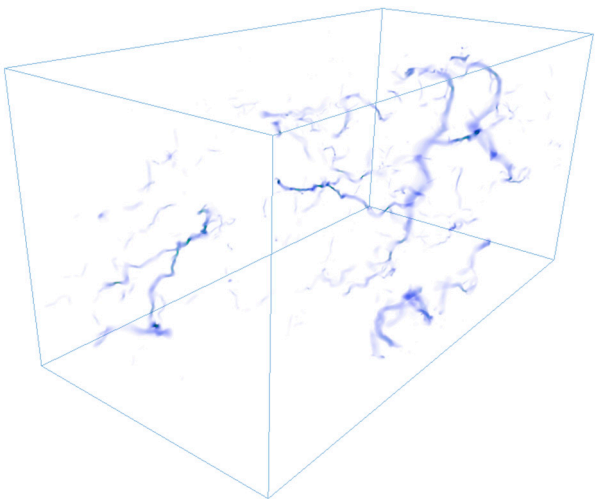

Figure 6. The velocity distribution computed in the pore space image of the Portland carbonate sample.

The average velocity throughout the medium is given over the simulation time for the different schemes in Figure 7. Most strikingly, the distribution method using floating point precision exhibits a large, systematic deviation from the double precision calculation of almost 50% and as such would give an overestimate of the permeability by the same amount. The perturbation method with float precision on the other hand matches the double-precision distribution calculation closely, yet requires considerably less memory and computing time to perform. Run on a Tesla K20 GPU, the float precision calculations required 27.8 s per 1000 time-steps and the double precision calculation required 48.3 s. In our sparse grid implementation, the array indices of each fluid node's 18 neighbours are also read in from the memory. Since these are stored as 4-byte integers in both float and double implementations, the speed up of 1.74x is in line with a memory-bandwidth-limited model.

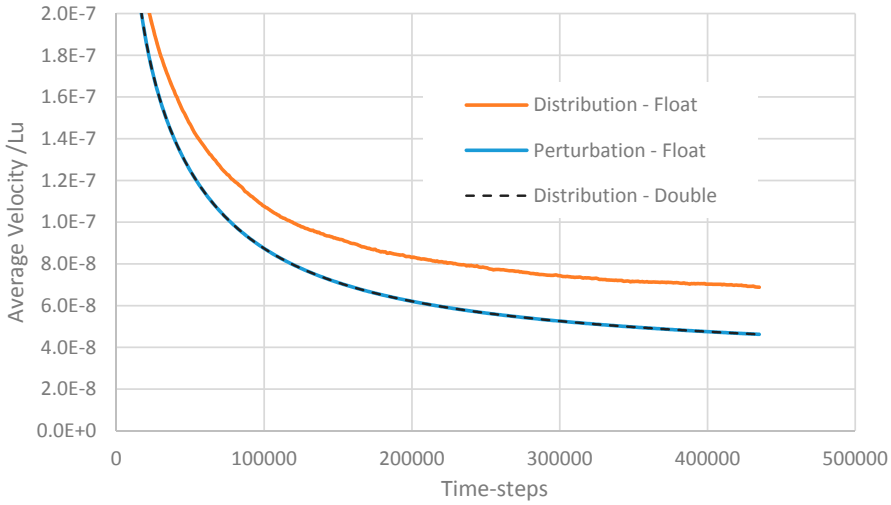

Figure 7. The average velocity throughout the Portland carbonate sample over the simulation. Points are sampled every 1000 time-steps.

4. Conclusions

We have shown how the simple transformation of the LB distribution function proposed by Skordos [5], does indeed lead to a considerable increase in computational precision when correctly implemented. Furthermore, the accuracy of this approach is no longer dependent on the magnitude of the velocity at the grid node. With application to flows in complex porous media, the modified scheme was shown to accurately compute the velocity field inside a heterogeneous pore-space when the average velocity became small. This means that it is no longer imperative to revert to using double precision arithmetic to achieve numerical stability and accuracy in simulations involving small grid velocities. Since the perturbation method incurs no extra computational burden, the computational efficiency is dramatically improved for these applications.

Acknowledgments: We gratefully acknowledge funding from the Qatar Carbonates and Carbon Storage Research Centre (QCCSRC), provided jointly by Qatar Petroleum, Shell, and Qatar Science and Technology Park. We would also like to thank three anonymous reviewers for their helpful comments and suggestions.

Author Contributions: Farrel Gray conceived and carried out the study, and prepared the text and images for the paper. Edo Boek gave critical review and made minor revisions to the text.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. He, X.; Luo, L.-S. Theory of the lattice Boltzmann method: From the Boltzmann equation to the lattice Boltzmann equation. *Phys. Rev. E* **1997**, *56*, 6811. [CrossRef]

2. Sukop, M.C.; Thorne, D.T., Jr. *Lattice Boltzmann Modeling: An Introduction for Geoscientists and Engineers*; Springer Publishing Company: Berlin, Germany, 2007.
3. Kang, Q.; Lichtner, P.C.; Viswanathan, H.S.; Abdel-Fattah, A.I. Pore Scale Modeling of Reactive Transport Involved in Geologic CO_2 Sequestration. *Transp. Porous Media* **2009**, *82*, 197–213. [CrossRef]
4. Bösch, F.; Chikatamarla, S.S.; Karlin, I.V. Entropic Multi-Relaxation Models for Turbulent Flows. *Phys. Rev. E* **2015**, *92*, 043309. [CrossRef] [PubMed]
5. Skordos, P. Initial and boundary conditions for the lattice Boltzmann method. *Phys. Rev. E* **1993**, *48*, 4823. [CrossRef]
6. FlowKit Ltd. *Palabos CFD*; FlowKit Ltd.: Lausanne, Switzerland, 2011.
7. Krause, M.J. OpenLB. 2014. Available online: http://optilb.com/openlb/ (accessed on 14 February 2016).
8. Dellar, P.J. Incompressible limits of lattice Boltzmann equations using multiple relaxation times. *J. Comput. Phys.* **2003**, *190*, 351–370. [CrossRef]
9. Crawshaw, J.P.; Boek, E.S. Multi-scale imaging and simulation of structure, flow and reactive transport for CO_2 storage and EOR in carbonate reservoirs. *Rev. Mineral. Geochem.* **2013**, *77*, 431–458. [CrossRef]
10. Yang, J.; Crawshaw, J.; Boek, E.S. Quantitative determination of molecular propagator distributions for solute transport in homogeneous and heterogeneous porous media using lattice Boltzmann simulations. *Water Resour. Res.* **2013**, *49*, 8531–8538. [CrossRef]
11. Blunt, M.J.; Bijeljic, B.; Dong, H.; Gharbi, O.; Iglauer, S.; Mostaghimi, P.; Paluszny, A.; Pentland, C. Pore-scale imaging and modelling. *Adv. Water Resour.* **2013**, *51*, 197–216. [CrossRef]
12. He, X.; Zou, Q.; Luo, L.; Dembo, M. Analytic solutions of simple flows and analysis of nonslip boundary conditions for the lattice Boltzmann BGK model. *J. Stat. Phys.* **1997**, *87*, 115–136. [CrossRef]
13. D'Humières, D. Multiple-relaxation-time lattice Boltzmann models in three dimensions. *Philos. Trans. A Math. Phys. Eng. Sci.* **2002**, *360*, 437–451. [CrossRef] [PubMed]
14. Lallemand, P.; Luo, L.-S. Theory of the lattice Boltzmann method: Dispersion, dissipation, isotropy, Galilean invariance, and stability. *Phys. Rev. E* **2000**, *61*, 6546. [CrossRef]
15. Pan, C.; Luo, L.-S.; Miller, C.T. An evaluation of lattice Boltzmann schemes for porous medium flow simulation. *Comput. Fluids* **2006**, *35*, 898–909. [CrossRef]
16. Ginzburg, I.; d'Humières, D. Multireflection boundary conditions for lattice Boltzmann models. *Phys. Rev. E* **2003**, *68*, 066614. [CrossRef] [PubMed]
17. Bouzidi, M.H.; Firdaouss, M.; Lallemand, P. Momentum transfer of a Boltzmann-lattice fluid with boundaries. *Phys. Fluids* **2001**, *13*, 3452–3459. [CrossRef]
18. Shah, S.; Gray, F.; Crawshaw, J.P.; Boek, E.S. Micro-computed tomography pore-scale study of flow in porous media: Effect of voxel resolution. *Adv. Water Resour.* **2015**. in press. [CrossRef]
19. Guo, Z.; Zheng, C.; Shi, B. Discrete lattice effects on the forcing term in the lattice Boltzmann method. *Phys. Rev. E* **2002**, *65*, 046308. [CrossRef] [PubMed]
20. Kuzmin, A.; Guo, Z.; Mohamad, A. Simultaneous incorporation of mass and force terms in the multi-relaxation-time framework for lattice Boltzmann schemes. *Philos. Trans. A Math. Phys. Eng. Sci.* **2011**, *369*, 2219–2227. [CrossRef] [PubMed]
21. Guo, Z.; Zheng, C. Analysis of lattice Boltzmann equation for microscale gas flows: Relaxation times, boundary conditions and the Knudsen layer. *Int. J. Comput. Fluid Dyn.* **2008**, *22*, 465–473. [CrossRef]
22. Jones, B.; Feng, Y. Effect of image scaling and segmentation in digital rock characterisation. *Comput. Part. Mech.* **2015**. [CrossRef]